

Wavelet analysis software developed for the paper:

*Measured changes in ocean surface roughness due to atmospheric
boundary layer rolls*

**By Steven A. Bailey
6/14/99**

Table of Contents

FILE FORMAT	3
MATLAB BATCH FILES	4
BATCH4.M.....	4
BATCH5.M.....	6
BATCH6.M.....	7
UXVSSRADAR1.M	8
UX_RADS_SRADAR.M	11
MATLAB FUNCTIONS.....	13
TABLE.M	13
MTF3.M.....	14
LONGEZ_QUICK_EXACT.M	15
REALOGRAM2.M	17
LONGEZ_QUICK_DENSITY.M	18
INTERPOLATE.M	21
SUN2PC.M	22
LONGEZ_QUICK_EXACT.M	22
EQUATIONS.....	24
VARIANCE OF AUTOSPECTRAL DENSITY (PSD).....	24
COEHERENCE ESTIMATE.....	25
MTF	26
WAVELET TO PSD SCALING	26
REFERENCES.....	27

File Format

The following file format is used in the V3, V4, and V5 directories.

Index	Variable	Description
0	Ux	Fluctuating wind component along-wind (% of mean)
1	Uy	Fluctuating wind component cross-wind (% of mean)
2	Uz	Fluctuating wind component vertical-wind (% of mean)
3	Sradar	Fluctuation of radar (with aircraft removed) (% of mean)
4	Rads	Fluctuation of inferred radar short slope variance (% of mean)
5	S2d	Fluctuation of 2D tilt variance (% of mean)
6	S1dx	Fluctuation of 1D cross-track tilt variance (% of mean)
7	S1dalong	Fluctuation of 1D along-track tilt variance (% of mean)
8	Ptang	Aircraft pointing angle (radians)
9	Altitude	Aircraft altitude (m)
10	Distance	Along-track distance (m)

Of special note is the variable 'Rads' of index 4. This variable is calculated by subtracting the 2D long wave tilt variance (S2d) from the averaged (400 m) total tilt variance of 'Sradar'. 'Rads' is then an estimate of the short wave tilt variance.

Matlab Batch Files

Batch4.m

The following is from the batch file: 'Batch4.m' found in the V4 directory. This batch file creates 25 text files which are each in a columnar format.

```
table( 'longez_samp.nov5.mab1121927.v4.bin.interp.mat', 'Rads',      'Ux', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1121927.v4.bin.interp.mat', 'Sradar',    'Ux', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1121927.v4.bin.interp.mat', 'Rads',      'Uz', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1121927.v4.bin.interp.mat', 'Sradar',    'Uz', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1121927.v4.bin.interp.mat', 'Sldalong',  'Ux', 1, 10, -9, -4, 8);

table( 'longez_samp.nov5.mab1121100.v4.bin.interp.mat', 'Rads',      'Ux', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1121100.v4.bin.interp.mat', 'Sradar',    'Ux', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1121100.v4.bin.interp.mat', 'Rads',      'Uz', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1121100.v4.bin.interp.mat', 'Sradar',    'Uz', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1121100.v4.bin.interp.mat', 'Sldalong',  'Ux', 1, 10, -9, -4, 8);

table( 'longez_samp.nov5.mab1120440.v4.bin.interp.mat', 'Rads',      'Ux', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1120440.v4.bin.interp.mat', 'Sradar',    'Ux', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1120440.v4.bin.interp.mat', 'Rads',      'Uz', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1120440.v4.bin.interp.mat', 'Sradar',    'Uz', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1120440.v4.bin.interp.mat', 'Sldalong',  'Ux', 1, 10, -9, -4, 8);

table( 'longez_samp.nov5.mab1114645.v4.bin.interp.mat', 'Rads',      'Ux', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1114645.v4.bin.interp.mat', 'Sradar',    'Ux', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1114645.v4.bin.interp.mat', 'Rads',      'Uz', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1114645.v4.bin.interp.mat', 'Sradar',    'Uz', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1114645.v4.bin.interp.mat', 'Sldalong',  'Ux', 1, 10, -9, -4, 8);

table( 'longez_samp.nov5.mab1113200.v4.bin.interp.mat', 'Rads',      'Ux', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1113200.v4.bin.interp.mat', 'Sradar',    'Ux', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1113200.v4.bin.interp.mat', 'Rads',      'Uz', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1113200.v4.bin.interp.mat', 'Sradar',    'Uz', 1, 10, -9, -4, 8);
table( 'longez_samp.nov5.mab1113200.v4.bin.interp.mat', 'Sldalong',  'Ux', 1, 10, -9, -4, 8);
```

One of the created files is:

File = longez_samp.nov5.mab1113200.v4.bin with Rads(x) vs. Ux(y)

Scale	Wavelength(m)	PSD_x	PSD_y	Coherence	Slope(real)	MTF
1	5120	+0.0175	+0.0253	+0.4560	+0.4389	+0.5251
2	4695	+0.0195	+0.0300	+0.3896	+0.3933	+0.4493
3	4305	+0.0266	+0.0312	+0.2101	+0.5307	+0.3872
4	3948	+0.0347	+0.0337	+0.4624	+0.6223	+0.6696
5	3620	+0.0363	+0.0338	+0.7034	+0.6524	+0.8824
6	3319	+0.0313	+0.0290	+0.7980	+0.6472	+0.9478
7	3044	+0.0233	+0.0221	+0.7573	+0.5812	+0.8895
8	2791	+0.0163	+0.0171	+0.6188	+0.4021	+0.7524
9	2560	+0.0124	+0.0147	+0.4716	+0.2431	+0.6312
10	2347	+0.0109	+0.0136	+0.3992	+0.3860	+0.5577
11	2152	+0.0113	+0.0139	+0.5475	+0.6825	+0.6130
12	1974	+0.0122	+0.0144	+0.6870	+0.8810	+0.6673
13	1810	+0.0128	+0.0155	+0.6726	+0.9661	+0.6605
14	1659	+0.0138	+0.0166	+0.5878	+0.9084	+0.6470
15	1522	+0.0147	+0.0161	+0.4874	+0.7333	+0.6368
16	1395	+0.0147	+0.0152	+0.3252	+0.5237	+0.5321
17	1280	+0.0137	+0.0144	+0.1590	+0.3189	+0.3638
18	1173	+0.0132	+0.0133	+0.0713	+0.2000	+0.2623
19	1076	+0.0146	+0.0112	+0.0514	+0.1631	+0.2701
20	987	+0.0149	+0.0091	+0.0449	+0.1309	+0.3024
21	905	+0.0137	+0.0082	+0.0469	+0.1345	+0.3263
22	829	+0.0118	+0.0087	+0.1121	+0.2432	+0.4542
23	761	+0.0102	+0.0093	+0.2290	+0.4010	+0.5633
24	697	+0.0088	+0.0092	+0.2778	+0.4818	+0.5621
25	640	+0.0074	+0.0086	+0.2345	+0.4704	+0.4768
26	586	+0.0061	+0.0075	+0.1605	+0.4302	+0.3560
27	538	+0.0049	+0.0065	+0.1137	+0.4279	+0.2632
28	493	+0.0039	+0.0058	+0.0995	+0.4659	+0.2134
29	452	+0.0032	+0.0055	+0.0640	+0.4201	+0.1505
30	414	+0.0031	+0.0053	+0.0269	+0.1608	+0.0936
31	380	+0.0031	+0.0049	+0.0331	-0.1099	+0.1110
32	348	+0.0032	+0.0047	+0.0353	-0.1987	+0.1293
33	320	+0.0034	+0.0046	+0.0377	-0.2258	+0.1498
34	293	+0.0035	+0.0044	+0.0615	-0.2853	+0.2037
35	269	+0.0033	+0.0040	+0.0929	-0.3624	+0.2444
36	246	+0.0028	+0.0038	+0.0935	-0.4178	+0.2142
37	226	+0.0022	+0.0037	+0.0475	-0.3500	+0.1268
38	207	+0.0019	+0.0035	+0.0091	-0.0582	+0.0503
39	190	+0.0019	+0.0034	+0.0308	+0.2539	+0.0955
40	174	+0.0020	+0.0034	+0.0589	+0.3811	+0.1370
41	160	+0.0019	+0.0035	+0.0508	+0.3554	+0.1251

Definitions for each column variable are:

Scale	Wavelet scale (index)
Wavelength	Actual wavelength (in meters) that scale is centered over
PSD_x	Power spectral density (auto-spectra) of 'x' variable. In the above file, the 'x' variable corresponds to 'Rads'. PSD (in this case) is computed by taking the mean power across each scale and then applying a correction. This correction removes extra energy as frequency increases due to the broadening of the wavelet bandpass as frequency increases.
PSD_y	Power spectral density (auto-spectra) of 'y' variable. In the above file, the 'y' variable corresponds to 'Ux'. PSD (in this case) is computed by taking the mean power across each scale and then applying a correction. This correction removes extra energy as frequency increases due to the broadening of the wavelet bandpass as frequency increases.

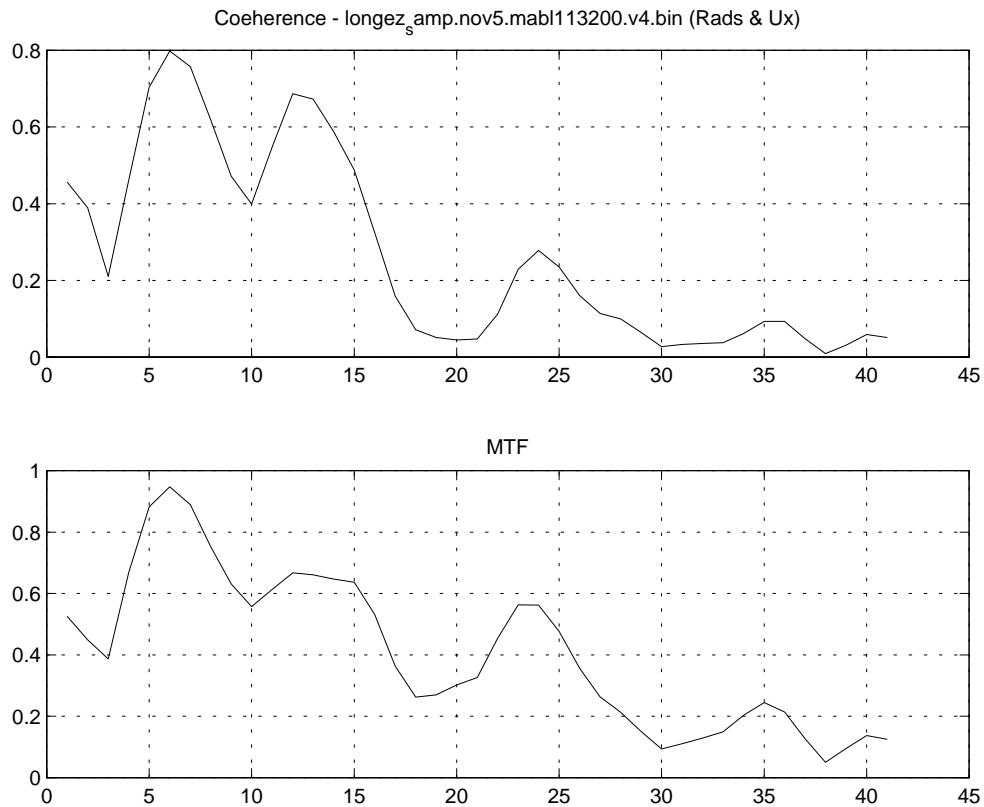
Coherence	A number ranging from 0 to 1 which indicates how 'similar' 2 variables are to each other. 0 means no coherence and 1 means perfect coherence. This function determines how the cross-spectral density of both variables is influenced by the product of the auto-spectral density of each individual variable.
Slope(real)	This is the slope of a line which is passed between the real components of 2 variables in the 'least squares' sense.
MTF	Modulation Transfer Function is similar to the coherence function. The difference here is that cross-spectral density of both variables is checked for modulation or coherence with the auto-spectral density of only one of the variables.

Batch5.m

The following is from the batch file: 'Batch5.m' found in the V4 directory. This batch file creates 5 figures (1 for each data file). Each figures contains 2 subplots. The top subplot is coherence and the bottom plot is MTF.

```
[SF,W1,W2] = mtf3( 'longez_samp.nov5.mab1121100.v4.bin.interp.mat', 'Rads', 'Ux', 1, 10, -9,
-4, 8);
print -dwin
[SF,W1,W2] = mtf3( 'longez_samp.nov5.mab1113200.v4.bin.interp.mat', 'Rads', 'Ux', 1, 10, -9,
-4, 8);
print -dwin
[SF,W1,W2] = mtf3( 'longez_samp.nov5.mab1114645.v4.bin.interp.mat', 'Rads', 'Ux', 1, 10, -9,
-4, 8);
print -dwin
[SF,W1,W2] = mtf3( 'longez_samp.nov5.mab1121927.v4.bin.interp.mat', 'Rads', 'Ux', 1, 10, -9,
-4, 8);
print -dwin
[SF,W1,W2] = mtf3( 'longez_samp.nov5.mab1120440.v4.bin.interp.mat', 'Rads', 'Ux', 1, 10, -9,
-4, 8);
print -dwin
```

Below is one of the five plots created using 'batch5.m'.



Batch6.m

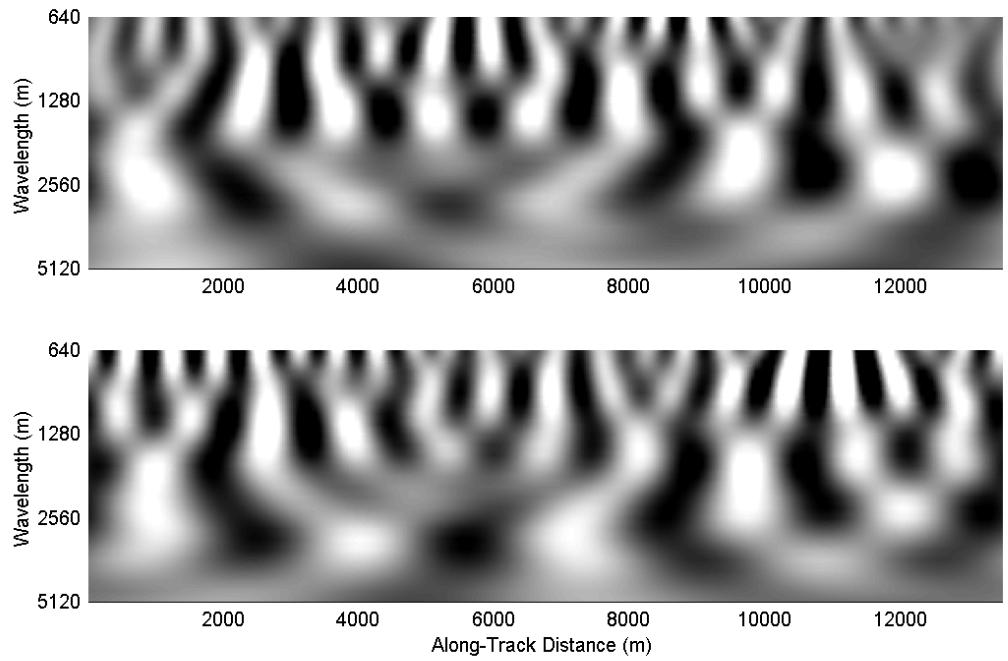
This batch file is just a longer version of 'Batch4.m'. Instead of comparing 5 variables per file, 'Batch6.m' compares 9 variables per file. 'Batch6.m' is used in both the V4 and V5 directories.

UxvsSradar1.m

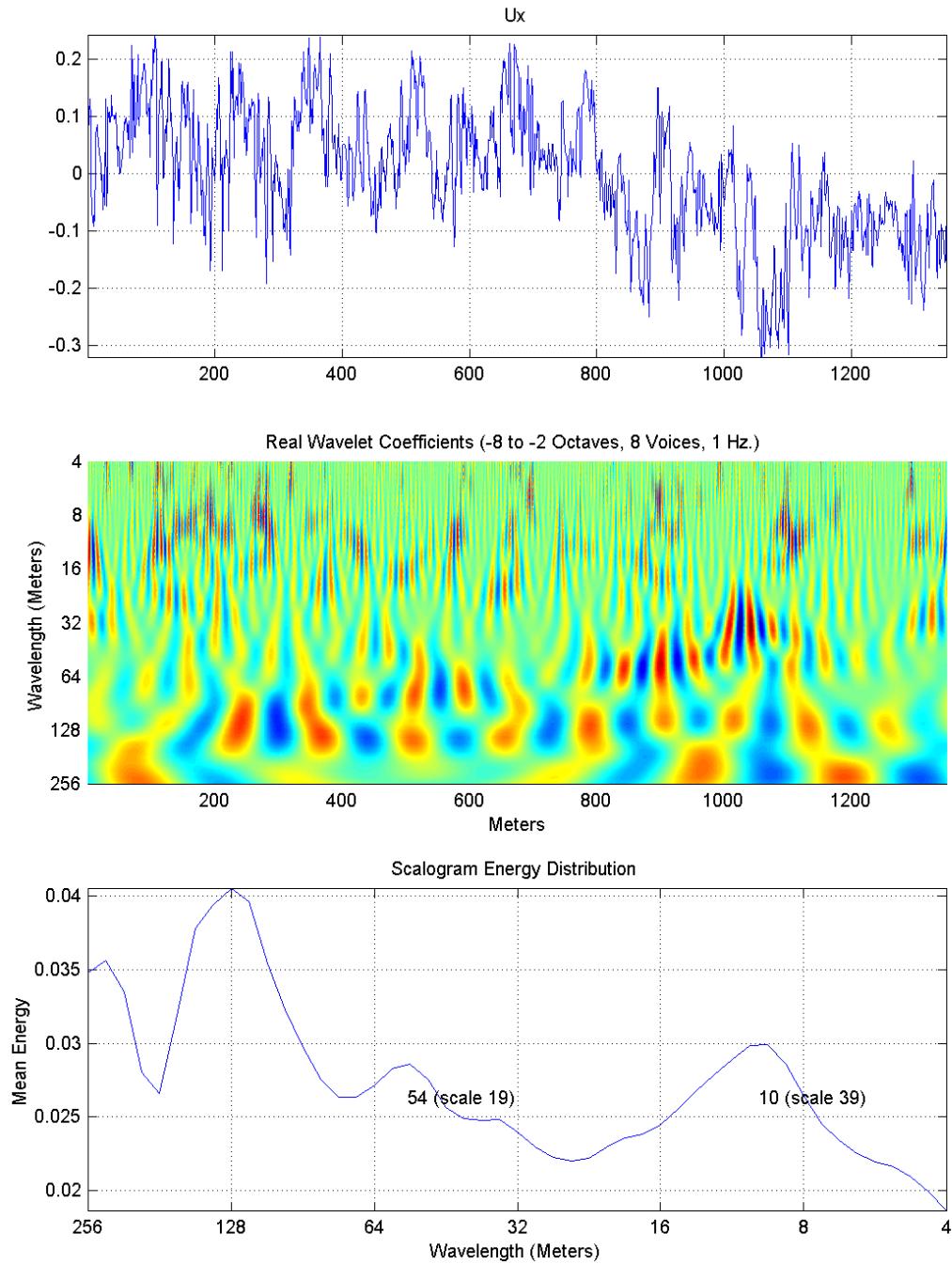
The following is from the batch file 'UxvsSradar1.m' found in the V3 directory.

```
load longez_samp.nov5.mabl121927.v3.bin.interp.mat  
[SF,W_Ux] = longez_quick_exact( Ux, 'Ux from mabl121927', 1, 10, -9, -6, 8);  
[SF,W_Sradar] = longez_quick_exact( Sradar, 'Sradar from mabl121927', 1, 10, -9, -6, 8);  
  
clf;  
subplot(4,1,1);  
Scalo = realogram2( W_Ux, SF, '', '', 'Wavelength (m)', 10);  
caxis([-0.04 .04]);  
  
subplot(4,1,2);  
Scalo = realogram2( W_Sradar, SF, '', 'Along-Track Distance (m)', 'Wavelength (m)', 10);  
caxis([-0.06 .06]);  
  
colormap(gray);
```

This batch file produces 1 figure composed of 2 subplots. The top plot shows the real wavelet components vs time for the variable 'Ux'. The bottom plot shows the real wavelet component vs time for the variable 'Sradar'. No titles are found because this figure will be imported elsewhere for publication.



Although the function 'longez_quick_exact.m' is used, the power of this function is not realized in this batch file. Typical graphical output from this function is of the form:



Ux_rads_sradar.m

The following is from the batch file 'ux_rads_sradar.m' found in the V3 directory.

```
omin      = -9;
omax      = -6;
xy_factor = 10;

load longez_samp.nov5.mabl121927.v3.bin.interp.mat; figure;
[SF,W_Ux] = longez_quick_density( Ux, '', 'k-', 1, xy_factor, omin, omax, 8); hold on;
[SF,W_Sradar] = longez_quick_density( Sradar, '', 'k:', 1, xy_factor, omin, omax, 8);
[SF,W_Rads] = longez_quick_density( Rads, '', 'k-.', 1, xy_factor, omin, omax, 8);
title('Mabl121927 Ux (--) 1/NRCS (- -) Rads (-.-)'');

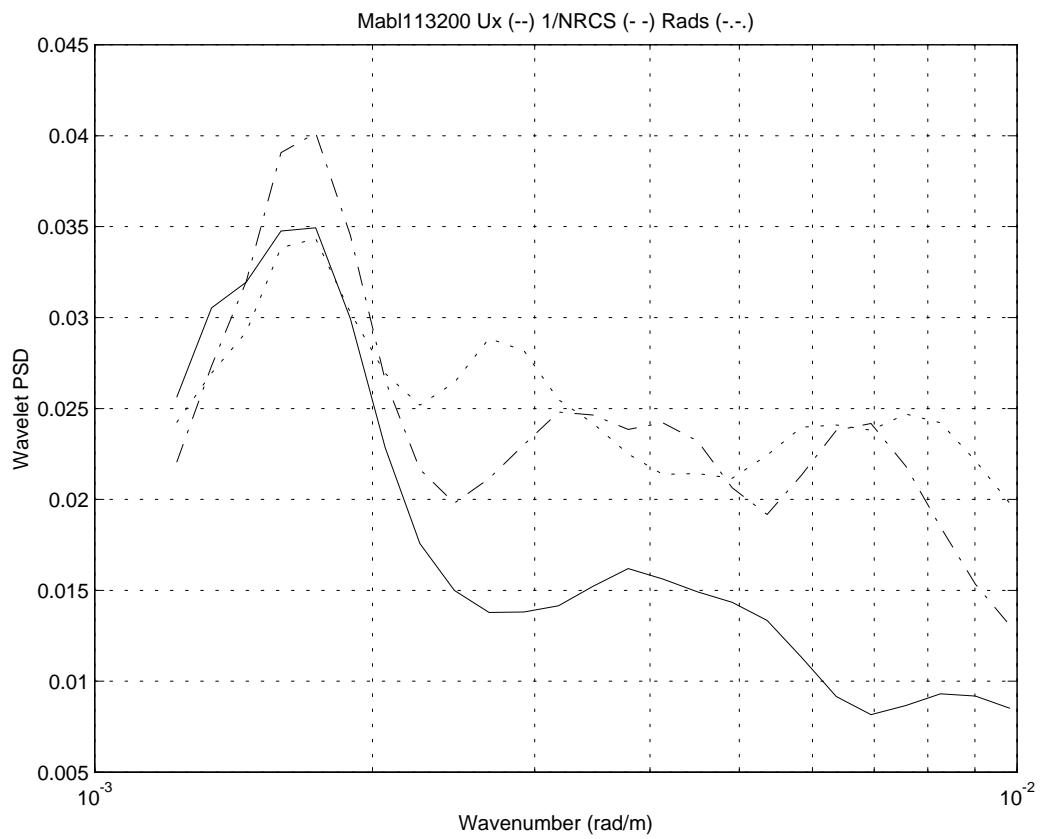
load longez_samp.nov5.mabl121100.v3.bin.interp.mat; figure;
[SF,W_Ux] = longez_quick_density( Ux, '', 'k-', 1, xy_factor, omin, omax, 8); hold on;
[SF,W_Sradar] = longez_quick_density( Sradar, '', 'k:', 1, xy_factor, omin, omax, 8);
[SF,W_Rads] = longez_quick_density( Rads, '', 'k-.', 1, xy_factor, omin, omax, 8);
title('Mabl121100 Ux (--) 1/NRCS (- -) Rads (-.-)'');

load longez_samp.nov5.mabl120440.v3.bin.interp.mat; figure;
[SF,W_Ux] = longez_quick_density( Ux, '', 'k-', 1, xy_factor, omin, omax, 8); hold on;
[SF,W_Sradar] = longez_quick_density( Sradar, '', 'k:', 1, xy_factor, omin, omax, 8);
[SF,W_Rads] = longez_quick_density( Rads, '', 'k-.', 1, xy_factor, omin, omax, 8);
title('Mabl120440 Ux (--) 1/NRCS (- -) Rads (-.-)'');

load longez_samp.nov5.mabl114645.v3.bin.interp.mat; figure;
[SF,W_Ux] = longez_quick_density( Ux, '', 'k-', 1, xy_factor, omin, omax, 8); hold on;
[SF,W_Sradar] = longez_quick_density( Sradar, '', 'k:', 1, xy_factor, omin, omax, 8);
[SF,W_Rads] = longez_quick_density( Rads, '', 'k-.', 1, xy_factor, omin, omax, 8);
title('Mabl114645 Ux (--) 1/NRCS (- -) Rads (-.-)'');

load longez_samp.nov5.mabl113200.v3.bin.interp.mat; figure;
[SF,W_Ux] = longez_quick_density( Ux, '', 'k-', 1, xy_factor, omin, omax, 8); hold on;
[SF,W_Sradar] = longez_quick_density( Sradar, '', 'k:', 1, xy_factor, omin, omax, 8);
[SF,W_Rads] = longez_quick_density( Rads, '', 'k-.', 1, xy_factor, omin, omax, 8);
title('Mabl113200 Ux (--) 1/NRCS (- -) Rads (-.-)'');
```

This batch file produces 5 figures which each contain an overplot of 3 variables. The variables are 'Ux', 'Sradar', and 'Rads'. Wavelet (corrected) PSD is calculated for each and are plotted vs. wavenumber. The following plot is an example of 1 of the figures.



Matlab Functions

Table.m

```

function table( filename, v1, v2, ns, xy_factor, omin, omax, nv )

%-----
%----- table( filename, v1, v2, ns, xy_factor, omin, omax, nv );
%-----%
%----- This function computes creates a table of scale, coherence, & slope of 2 variables.
%-----%
%----- filename - Matlab filename (in PC longez format)
%----- v1 - Variable 1
%----- v2 - Variable 2
%----- ns - Sampling rate (Hz)
%----- xy_factor - X-Y factor
%----- omin - Octave min
%----- omax - Octave max
%----- nv - Number of voices
%-----%
%----- SAB NASA 5/6/99
%-----%

str = sprintf('load %s', filename );
eval(str);

str = sprintf('[SF, W1] = longez_direct_exact( %s, %d, %d, %d, %d, %d );', v1, ns, xy_factor,
omin, omax, nv );
eval(str);

str = sprintf('[SF, W2] = longez_direct_exact( %s, %d, %d, %d, %d, %d );', v2, ns, xy_factor,
omin, omax, nv );
eval(str);

%-----
% MTF section
%-----

P      = fft(W1);           % Received Power from Radar
S      = fft(W2);           % Ux wind component
GPS   = mean(P.*conj(S));  % Cross-spectrum
GSS   = mean(S.*conj(S));  % Auto-spectrum (PSD)

M     = abs(GPS./GSS);

%-----

s      = size(SF);
s2    = size(W1);
c     = zeros(s(1),1);
coh   = zeros(s(1),1);
coefs_r = zeros(s(1),2);
coefs_i = zeros(s(1),2);
coefs_m = zeros(s(1),2);

for i=1:s(1)
    c(i)      = xcorr(real(W1(:,i)),real(W2(:,i)),0,'coeff');          % Correlation function at
                                                                     % at scale 'i'
    a          = mean(W1(:,i));                                         % Working variables
    b          = mean(W2(:,i));
    abcc     = (W1(:,i)'.*W2(:,i))/s2(1);
    aacc     = (W1(:,i)'.*W1(:,i))/s2(1);
    bbcc     = (W2(:,i)'.*W2(:,i))/s2(1);

    coh(i)    = (abs(abcc)^2) / (aacc*bbcc);                         % Coherence function at
                                                                     % scale 'i'

    coefs_r(i,:) = polyfit( real(W1(:,i)), real(W2(:,i)), 1 );        % Fit slope between W1 & W2
end

```

```

coefs_i(i,:) = polyfit( imag(W1(:,i)), imag(W2(:,i)), 1 ); % Fit slope between W1 & W2
coefs_m(i,:) = polyfit( abs(W1(:,i)), abs(W2(:,i)), 1 ); % Fit slope between W1 & W2
end

% Wavelet power spectral density (PSD) correction

psd_v1 = 10.^ log10( mean(abs(W1))' ) - .5*log10(s2(1)./SF(:,2)) + log10(1.5) ;
psd_v2 = 10.^ log10( mean(abs(W2))' ) - .5*log10(s2(1)./SF(:,2)) + log10(1.5) ;

z = [ SF(:,1)'; SF(:,2)'*xy_factor; psd_v1'; psd_v2'; coh'; coefs_r(:,1)'; M ];

str = sprintf('%s.%s_vs_%s.txt', filename, v1, v2 );
fid = fopen( str, 'w' );
fprintf(fid, 'File = %s with %s(x) vs. %s(y)\n\n', filename, v1, v2 );
fprintf(fid, ' Scale Wavelength(m) PSD_x PSD_y Coherence Slope(real) MTF\n');
fprintf(fid, '-----\n');
fprintf(fid, '%4d %5d %+5.4f %+5.4f %+5.4f %+5.4f %+5.4f\n', z );
fclose(fid);

```

Mtf3.m

```

function [SF, W1, W2] = mtf3( filename, v1, v2, ns, xy_factor, omin, omax, nv )

%-----%
% [SF, W1, W2] = mtf3( filename, v1, v2, ns, xy_factor, omin, omax, nv );
%-----%
% This function computes the MTF (modulation transfer function) of the said
% variables.
%-----%
% filename - Matlab filename (in PC longez format)
% v1 - Variable 1
% v2 - Variable 2
% ns - Sampling rate (Hz)
% xy_factor - X-Y factor
% omin - Octave min
% omax - Octave max
% nv - Number of voices
%-----%
% SAB NASA 5/14/99
%-----%

str = sprintf('load %s', filename );
eval(str);

str = sprintf('[SF, W1] = longez_direct_exact( %s, %d, %d, %d, %d, %d );', v1, ns, xy_factor,
omin, omax, nv );
eval(str);

str = sprintf('[SF, W2] = longez_direct_exact( %s, %d, %d, %d, %d, %d );', v2, ns, xy_factor,
omin, omax, nv );
eval(str);

s = size(SF);
s2 = size(W1);

%-----%
% MTF section
%-----%

P = fft(W1); % Received Power from Radar
S = fft(W2); % Ux wind component
GPS = mean(P.*conj(S)); % Cross-spectrum
GSS = mean(S.*conj(S)); % Auto-spectrum (PSD)

M = abs(GPS./GSS);

%-----%
% Coherence section
%-----%

```

```

coeh      = zeros(s(1),1);

for i=1:s(1)
    a          = mean(W1(:,i));                      % Working variables
    b          = mean(W2(:,i));
    abcc      = (W1(:,i)'*W2(:,i))/s2(1);
    aacc      = (W1(:,i)'*W1(:,i))/s2(1);
    bbcc      = (W2(:,i)'*W2(:,i))/s2(1);

    coeh(i)   = (abs(abcc)^2) / (aacc*bbcc);        % Coherence function at scale 'i'
end

%-----

str = sprintf('Coherence - %s (Rads & Ux)', filename );

subplot(2,1,1);
plot(coeh);
grid;
title(str);

subplot(2,1,2);
plot(M);
grid;
title('MTF');

```

Longez_quick_exact.m

```

function [SF, W] = longez_quick_exact(x, toptitle, ns, xy_factor, omin, omax, nv)

%-----
% [SF, W] = longez_quick_exact( x, toptitle, ns, xy_factor, omin, omax, nv );
%-----
% Octave min is related to the lowest frequency such as 2^omin
% For example Octave min = -2 --> .25 Hz
% Octave max for the Highest frequency 2^omax
% Even though the concept is pretty wide for scale analysis these
% frequencies will be related to the peak in the Fourier domain
%-----
% The above description is from the original written by Bertrand Chapron from
% 'mtrans.m'. I have modified this function in the follow ways:
%
% I added 'text' info. on the command line.
% I added a closed form function for determining 'wavelength' vs 'scale'.
% I output the Wavelet coeffs. and 2 vectors of 'scale' and 'wavelength'.
%
% This version uses function 'realogram' and makes a third plot of
% scalogram energy distribution. This version also alters Bertrands
% code slightly so we start and stop exactly on the given octaves.
%
% x         - Original signal (M x 1)
% toptitle - Text string of top plot title
% ns        - Sampling rate (Hz)
% xy_factor - X-Y factor
% omin      - Octave min
% omax      - Octave max
% nv        - Number of voices
%-----
% SAB       NASA           5/3/99
%-----


ntot      = abs(omax-omin)*nv+1;

i0      = 1;
tp      = size(x);
n       = tp(1);

f      = fft( x(i0:i0+n-1) );
f      = f(1:floor(n/2)+1);
f(1) = f(1)/2;

for i=1:ntot          % Create copies of FFT of data for each scale

```

```

        F(:,i)=f;
    end

k=1;
%k_trans=menu(' Time-Frequency Representation : ','Wavelet Transform ','Short Time Fourier
Transform');

k_trans = 1;                                % Hardwire for Longez work
k_choice = 1;                                % Hardwire for Longez work

if(k_trans==1)
% k_choice=menu(' Wavelet Choice : ','Morlet','Mexican Hat');

if(k_choice==1)
    for i=omin:omax-1 % Create Morlet Wavelets for each scale (in freq. domain)
        for j=0:nv-1
            a(k)=2^(i+j/nv);
            w(:,k)=momo(1/a(k),n,ns);
            k=k+1;
        end
        a(k) = 2^omax;
        w(:,k)=momo(1/a(k),n,ns);
    end
end

if(k_choice==2)
    for i=omin:omax    % Creat Mexican Hat Wavelets for each scale (in freq. domain)
        for j=0:nv-1
            a(k)=2^(i+j/nv);
            w(:,k)=mxmx(1/a(k),n,ns);
            k=k+1;
        end
    end
end
end

if(k_trans==2)
    k_choice=menu(' Waveform Choice : ','Gabor Waveform','Modulated Hanning');

if(k_choice==1)
    for i=omin:omax
        for j=0:nv-1
            a(k)=2^(i+j/nv);
            w(:,k)=mfmf(a(k),n,ns);
            k=k+1;
        end
    end
end
if(k_choice==2)
    nh=input(' Window length : ');
    for i=omin:omax
        for j=0:nv-1
            a(k)=2^(i+j/nv);
            w(:,k)=mhmh(a(k),n,ns,nh);
            k=k+1;
        end
    end
end
end

W=ifft(2*F.*w,n);      % Convolve Wavelet with Signal (in freq. domain), then
                        % deconvolve back into time domain giving wavelet coefficients

s = size(W);
SF = zeros(s(2),2);

for i=1:s(2)           % Produce 2 vectors of scale & wavelength
    SF(i,1) = i;
    SF(i,2) = 2^( abs(omin) - (i - 1)*(1/nv) ) * ns;
end

SF(:,2) = SF(:,2).*10; % Reduce to 1 digit of precision
SF(:,2) = floor(SF(:,2));
SF(:,2) = SF(:,2)./10;

```

```

maxx = (n-1)*xy_factor;
index = 0:xy_factor:maxx;

clf;
subplot(3,1,1); % Plot 1st subplot
plot(index,x);
title(toptitle);
axis([ 1 maxx min(x) max(x) ]);
grid;

subplot(3,1,2); % Plot 2nd subplot
Scalo = realogram2( W, SF, '', 'Meters', 'Wavelength (Meters)', xy_factor );
str = sprintf('Real Wavelet Coefficients (%d to %d Octaves, %d Voices, %d Hz.)', omin, omax,
nv, ns );
title(str);

subplot(3,1,3); % Plot 3rd subplot
c = mean(Scalo);
plot( c );
axis([ 1 s(2) min(c) max(c) ]);
range = abs( omin - omax ) + 1;

j = abs(omin);
for i=1:range
    t(i,1) = find( SF(:,2) == 2^j );
    t(i,2) = 2^j*xy_factor;
    j = j - 1;
end

set( gca, 'xtick', t(:,1) );
set( gca, 'xticklabel', t(:,2) );

xlabel('Wavelength (Meters)');
ylabel('Mean Energy');
title('Scalogram Energy Distribution');
grid;

g = ginput;
sg = size(g);

if ( sg(1) > 0 )
    for i=1:sg(1)
        str = sprintf('%d (scale %d)', round(SF( round(g(i,1)), 2 ) * xy_factor), SF(
round(g(i,1)), 1 ) );
        text( g(i,1), g(i,2), str );
    end
end

```

Realogram2.m

```

function Scalo = realogram( W, SF, t_title, x_label, y_label, xy_factor )

%-----
% Scalo = realogram( W, SF, t_title, x_label, y_label, xy_factor );
%-----
% This function takes the following input:
%
% W          - Wavelet coefficients
% SF         - Scale to Freq. Vectors
% t_title    - Title for plot
% x_label    - Xlabel for plot
% y_label    - Ylabel for plot
% xy_factor - Multiply this factor by 'x' & 'y' axis
%
% and creates a realogram (only real component) with Wavelength on y-axis
%-----
% SAB        NASA        5/3/99
%-----

s = size(W);      % Get size of W
div = round(s(2)/4); % Calculate tick divisor

```

```

maxx = (s(1)-1)*xy_factor;
index = 0:xy_factor:maxx;

Scalo = sqrt( real(W).^2 + imag(W).^2 );
surf(index, 1:s(2), real(W)');
view(0,90);
shading interp;

axis([1 maxx 1 s(2)]); % Fill figure with image

omin = log2( SF(1,2) );
omax = log2( floor( SF(s(2),2) ) );
range = abs( omin - omax )+1;

j = abs(omin);
for i=1:range
    t(i,1) = find( SF(:,2) == 2^j );
    t(i,2) = 2^j*xy_factor;
    j = j - 1;
end

set( gca, 'ytick', t(:,1) );
set( gca, 'yticklabel', t(:,2) );

title(t_title);
xlabel(x_label);
ylabel(y_label);

```

Longez_quick_density.m

```

function [SF, W] = longez_quick_exact3(x, toptitle, plottype, ns, xy_factor, omin, omax, nv)

%-----%
% [SF, W] = longez_quick_exact3( x, toptitle, plottype, ns, xy_factor, omin, omax, nv );
%-----%
% Octave min is related to the lowest frequency such as 2^omin
% For example Octave min = -2 --> .25 Hz
% Octave max for the Highest frequency 2^omax
% Even though the concept is pretty wide for scale analysis these
% frequencies will be related to the peak in the Fourier domain
%-----%
% The above description is from the original written by Bertrand Chapron from
% 'mtrans.m'. I have modified this function in the follow ways:
%-----%
% I added 'text' info. on the command line.
% I added a closed form function for determining 'wavelength' vs 'scale'.
% I output the Wavelet coeffs. and 2 vectors of 'scale' and 'wavelength'.
%-----%
% This version uses function 'realogram' and makes a third plot of
% scalogram energy distribution. This version also alters Bertrands
% code slightly so we start and stop exactly on the given octaves.
%-----%
% x          - Original signal (M x 1)
% toptitle   - Text string of top plot title
% plottype   - Text defining plot line type
% ns         - Sampling rate (Hz)
% xy_factor - X-Y factor
% omin       - Octave min
% omax       - Octave max
% nv         - Number of voices
%-----%
% SAB        NASA           5/3/99
%-----%

ntot      = abs(omax-omin)*nv+1;

i0      = 1;
tp      = size(x);
n      = tp(1);

f      = fft( x(i0:i0+n-1) );
f      = f(1:floor(n/2)+1);
f(1) = f(1)/2;

```

```

for i=1:ntot          % Create copies of FFT of data for each scale
    F(:,i)=f;
end

k=1;
%k_trans=menu(' Time-Frequency Representation : ','Wavelet Transform ','Short Time Fourier
Transform');

k_trans = 1;           % Hardwire for Longez work
k_choice = 1;          % Hardwire for Longez work

if(k_trans==1)
% k_choice=menu(' Wavelet Choice : ','Morlet','Mexican Hat');

    if(k_choice==1)
        for i=oamin:omax-1 % Create Morlet Wavelets for each scale (in freq. domain)
            for j=0:nv-1
                a(k)=2^(i+j/nv);
                w(:,k)=momo(1/a(k),n,ns);
                k=k+1;
            end
        end
        a(k) = 2^omax;
        w(:,k)=momo(1/a(k),n,ns);
    end

    if(k_choice==2)
        for i=oamin:omax    % Creat Mexican Hat Wavelets for each scale (in freq. domain)
            for j=0:nv-1
                a(k)=2^(i+j/nv);
                w(:,k)=mxmx(1/a(k),n,ns);
                k=k+1;
            end
        end
    end
end

if(k_trans==2)
    k_choice=menu(' Waveform Choice : ','Gabor Waveform','Modulated Hanning');

    if(k_choice==1)
        for i=oamin:omax
            for j=0:nv-1
                a(k)=2^(i+j/nv);
                w(:,k)=mfmf(a(k),n,ns);
                k=k+1;
            end
        end
    end

    if(k_choice==2)
        nh=input(' Window length : ');
        for i=oamin:omax
            for j=0:nv-1
                a(k)=2^(i+j/nv);
                w(:,k)=mhmh(a(k),n,ns,nh);
                k=k+1;
            end
        end
    end
end

W=ifft(2*F.*w,n);      % Convolve Wavelet with Signal (in freq. domain), then
                        % deconvolve back into time domain giving wavelet coefficients

s = size(W);
SF = zeros(s(2),2);

for i=1:s(2)           % Produce 2 vectors of scale & wavelength
    SF(i,1) = i;
    SF(i,2) = 2^( abs(oamin) - (i - 1)*(1/nv) ) * ns;
end

SF(:,2) = SF(:,2).*10; % Reduce to 1 digit of precision
SF(:,2) = floor(SF(:,2));

```

```

SF(:,2) = SF(:,2)./10;

Scalo = sqrt( real(W).^2 + imag(W).^2 );

% Wavelet PSD correction
c = log10(mean(Scalo')) - .5*log10(n./SF(:,2)) + log10(1.5);
semilogx( (2*pi)./(SF(:,2)*xy_factor), 10.^c, plottype );
%axis([ 10^-3 10^-1 0 .025 ] );

xlabel('Wavenumber (rad/m)');
ylabel('Wavelet PSD');
grid;

```

Interpolate.m

```
%-----  
% This function reads in a binary, PC file in the current longez  
% format. It then normalizes fields by distance using linear  
% interpolation.  
%-----  
% SAB      NASA      4/14/99  
%-----  
  
function interpolate( filename )  
  
M          = sun2pc(filename);  
s          = size(M);  
  
Ux          = M(1:11:s(1));  
Uy          = M(2:11:s(1));  
Uz          = M(3:11:s(1));  
Sradar     = M(4:11:s(1));  
Rads        = M(5:11:s(1));  
S2d         = M(6:11:s(1));  
S1dx        = M(7:11:s(1));  
S1dalong   = M(8:11:s(1));  
Ptang       = M(9:11:s(1));  
Altitude    = M(10:11:s(1));  
Distance    = M(11:11:s(1)).*1000;  
  
s          = size(Distance);  
x          = (1:Distance(s(1))');  
  
Ux_interp   = interp1(Distance,Ux,x,'linear');  
Uy_interp   = interp1(Distance,Uy,x,'linear');  
Uz_interp   = interp1(Distance,Uz,x,'linear');  
Sradar_interp = interp1(Distance,Sradar,x,'linear');  
Rads_interp  = interp1(Distance,Rads,x,'linear');  
S2d_interp   = interp1(Distance,S2d,x,'linear');  
S1dx_interp  = interp1(Distance,S1dx,x,'linear');  
S1dalong_interp = interp1(Distance,S1dalong,x,'linear');  
Ptang_interp  = interp1(Distance,Ptang,x,'linear');  
Altitude_interp = interp1(Distance,Altitude,x,'linear');  
Distance_interp = interp1(Distance,Distance,x,'linear');  
  
Ux          = Ux_interp(1:10:size(Ux_interp));  
Uy          = Uy_interp(1:10:size(Uy_interp));  
Uz          = Uz_interp(1:10:size(Uz_interp));  
Sradar     = Sradar_interp(1:10:size(Sradar_interp));  
Rads        = Rads_interp(1:10:size(Rads_interp));  
S2d         = S2d_interp(1:10:size(S2d_interp));  
S1dx        = S1dx_interp(1:10:size(S1dx_interp));  
S1dalong   = S1dalong_interp(1:10:size(S1dalong_interp));  
Ptang       = Ptang_interp(1:10:size(Ptang_interp));  
Altitude    = Altitude_interp(1:10:size(Altitude_interp));  
Distance    = Distance_interp(1:10:size(Distance_interp));  
  
clear Ux_interp Uy_interp Uz_interp Sradar_interp;  
clear Rads_interp S2d_interp S1dx_interp;  
clear S1dalong_interp Ptang_interp;  
clear Altitude_interp Distance_interp;  
clear s x M;  
  
str = sprintf('save %s.interp.mat', filename );  
eval(str);
```

Sun2pc.m

```
%-----  
% This function reads in a Sun produced binary file and produces  
% a PC/Matlab readable array. Sun produces Big Endian and  
% PC produces Little Endian. All we need to do is invert or  
% mirror Sun bits to PC bits. IE. - bit 32 becomes bit 1 and  
% vice versa. The only way I could figure out how to do this  
% is to read in each byte one at a time, and then write them  
% out in reversed order 4 bytes at a time. I then read them  
% back in and they convert directly to floats.  
%-----  
% SAB      NASA      12/11/98  
%-----  
  
function M = sun2pc( filename )  
  
fid = fopen( filename, 'r' );           % Read in 1 char at a time  
[A, Count] = fread( fid, 'uchar' );  
fclose( fid );  
  
fid = fopen( 'junk.bin', 'w' );          % Write them back out reversed  
for (i=4:4:Count)  
    fwrite( fid, A(i:-1:i-3), 'uchar' );  
end  
fclose( fid );  
  
fid = fopen( 'junk.bin', 'r' );           % Read back in and convert to float  
[M, Count] = fread( fid, 'float' );  
fclose( fid );
```

Longez_quick_exact.m

```
function [SF, W] = longez_quick_exact(x, toptitle, ns, xy_factor, omin, omax, nv)  
  
%-----  
% [SF, W] = longez_quick_exact( x, toptitle, ns, xy_factor, omin, omax, nv );  
%-----  
% Octave min is related to the lowest frequency such as 2^omin  
% For example Octave min = -2 --> .25 Hz  
% Octave max for the Highest frequency 2^omax  
% Even though the concept is pretty wide for scale analysis these  
% frequencies will be related to the peak in the Fourier domain  
%-----  
% The above description is from the original written by Bertrand Chapron from  
% 'mtrans.m'. I have modified this function in the follow ways:  
%  
% I added 'text' info. on the command line.  
% I added a closed form function for determining 'wavelength' vs 'scale'.  
% I output the Wavelet coeffs. and 2 vectors of 'scale' and 'wavelength'.  
%  
% This version uses function 'realogram' and makes a third plot of  
% scalogram energy distribution. This version also alters Bertrands  
% code slightly so we start and stop exactly on the given octaves.  
%  
% x      - Original signal (M x 1)  
% toptitle - Text string of top plot title  
% ns      - Sampling rate (Hz)  
% xy_factor - X-Y factor  
% omin    - Octave min  
% omax    - Octave max  
% nv      - Number of voices  
%-----  
% SAB      NASA      5/3/99  
%-----  
  
ntot      = abs(omax-omin)*nv+1;
```

```

i0    = 1;
tp    = size(x);
n     = tp(1);

f     = fft( x(i0:i0+n-1) );
f     = f(1:floor(n/2)+1);
f(1) = f(1)/2;

for i=1:ntot           % Create copies of FFT of data for each scale
    F(:,i)=f;
end

k=1;
%k_trans=menu(' Time-Frequency Representation : ','Wavelet Transform ','Short Time Fourier
Transform');

k_trans = 1;                % Hardwire for Longez work
k_choice = 1;                % Hardwire for Longez work

if(k_trans==1)
% k_choice=menu(' Wavelet Choice : ','Morlet','Mexican Hat');

if(k_choice==1)
    for i=oamin:omax-1 % Create Morlet Wavelets for each scale (in freq. domain)
        for j=0:nv-1
            a(k)=2^(i+j/nv);
            w(:,k)=momo(1/a(k),n,ns);
            k=k+1;
        end
        a(k) = 2^omax;
        w(:,k)=momo(1/a(k),n,ns);
    end
end

if(k_choice==2)
    for i=oamin:omax    % Creat Mexican Hat Wavelets for each scale (in freq. domain)
        for j=0:nv-1
            a(k)=2^(i+j/nv);
            w(:,k)=mxmx(1/a(k),n,ns);
            k=k+1;
        end
    end
end
end

if(k_trans==2)
k_choice=menu(' Waveform Choice : ','Gabor Waveform','Modulated Hanning');

if(k_choice==1)
    for i=oamin:omax
        for j=0:nv-1
            a(k)=2^(i+j/nv);
            w(:,k)=mfmf(a(k),n,ns);
            k=k+1;
        end
    end
end

if(k_choice==2)
    nh=input(' Window length : ');
    for i=oamin:omax
        for j=0:nv-1
            a(k)=2^(i+j/nv);
            w(:,k)=mhmh(a(k),n,ns,nh);
            k=k+1;
        end
    end
end
end

W=ifft(2*F.*w,n);      % Convolve Wavelet with Signal (in freq. domain), then
                        % deconvolve back into time domain giving wavelet coefficients

s = size(W);
SF = zeros(s(2),2);

```

```

for i=1:s(2)           % Produce 2 vectors of scale & wavelength
    SF(i,1) = i;
    SF(i,2) = 2^( abs(omin) - (i - 1)*(1/nv) ) * ns;
end

SF(:,2) = SF(:,2).*10; % Reduce to 1 digit of precision
SF(:,2) = floor(SF(:,2));
SF(:,2) = SF(:,2)./10;

maxx = (n-1)*xy_factor;
index = 0:xy_factor:maxx;

clf;
subplot(3,1,1);          % Plot 1st subplot
plot(index,x);
title(toptitle);
axis([ 1 maxx min(x) max(x) ]);
grid;

subplot(3,1,2);          % Plot 2nd subplot
Scalo = realogram2( W, SF, '', 'Meters', 'Wavelength (Meters)', xy_factor );
str = sprintf('Real Wavelet Coefficients (%d to %d Octaves, %d Voices, %d Hz.)', omin, omax,
nv, ns );
title(str);

subplot(3,1,3);          % Plot 3rd subplot
c = mean(Scalo);
plot( c );
axis([ 1 s(2) min(c) max(c) ]);
range = abs( omin - omax ) + 1;

j = abs(omin);
for i=1:range
    t(i,1) = find( SF(:,2) == 2^j );
    t(i,2) = 2^j*xy_factor;
    j = j - 1;
end

set( gca, 'xtick', t(:,1) );
set( gca, 'xticklabel', t(:,2) );

xlabel('Wavelength (Meters)');
ylabel('Mean Energy');
title('Scalogram Energy Distribution');
grid;

g = ginput;
sg = size(g);

if ( sg(1) > 0 )
    for i=1:sg(1)
        str = sprintf('%d (scale %d)', round(SF( round(g(i,1)), 2 ) * xy_factor), SF(
round(g(i,1)), 1 ) );
        text( g(i,1), g(i,2), str );
    end
end

```

Equations

Variance of autospectral density (PSD)

$$Var\left[\hat{G}_{xx}(f)\right] \approx \frac{G_{xx}^2(f)}{B_e T} \quad (8.143)$$

Where:

$G_{xx}^2(f)$ is the squared autospectral density of $x(t)$

B_e is the bandwidth of a given wavelet passband centered at f in units of 1/10 meters

T is the length between samples (10 meters)

Coherence estimate

$$\hat{\gamma}_{xy}^2 = \frac{|\hat{G}_{xy}(f)|^2}{\hat{G}_{xx}(f)\hat{G}_{yy}(f)} \quad (11.146)$$

Where:

$|\hat{G}_{xy}(f)|^2$ is the squared absolute value of the cross-spectral density estimate between $x(t)$ and $y(t)$

$\hat{G}_{xx}(f)$ is the autospectral density estimate of $x(t)$

$\hat{G}_{yy}(f)$ is the autospectral density estimate of $y(t)$

The following equation defines the normalized random error of the coherence function:

$$\varepsilon = \frac{\sqrt{2}[1 - \gamma_{xy}^2(f)]}{|\gamma_{xy}(f)|\sqrt{n}} \quad (\text{Table 9.6})$$

Where:

$\gamma_{xy}^2(f)$ is the coherence function

$|\gamma_{xy}(f)|$ is the absolute value of the square root of the coherence function.

\sqrt{n} is the square root of the number of ensemble averages that were made for this estimate.
In our case, $n = 1$

MTF

$$M(f) = \frac{\hat{G}_{xy}(f)}{\hat{G}_{yy}(f)} \quad (2)$$

$\hat{G}_{xy}(f)$ is the cross-spectral density estimate between $x(t)$ and $y(t)$

$\hat{G}_{yy}(f)$ is the autospectral density estimate of $y(t)$

There is no factor of mean received power used because the radar return is a zero mean process in this case.

Wavelet to PSD scaling

The Morlet Wavelet is simply a Gaussian bandpass filter (in the freq. domain). For each scale (freq.) in question, this filter is convolved with the FFT of our data. As the scale (freq.) increases, the bandpass of the filter widens. Likewise as the scale (freq.) decreases, the bandpass of the filter narrows. At frequencies approaching DC, our bandpass approaches an impulse.

This Wavelet filtering mechanism is a power of 2 based. Scales increase or decrease by a power of 2. Likewise, our filter bandpass follows suite. This convention allows for simple reconstruction of our original signal as the sum of the real components across all scales. This occurs when there are 8 or more octaves per scale. Unfortunately, the described convention biases our power spectral density estimate. This bias is removed with the following equation:

$$PSD = 10^{(\log_{10}(M) - .5\log_{10}(F) + \log_{10}(1.5))}$$

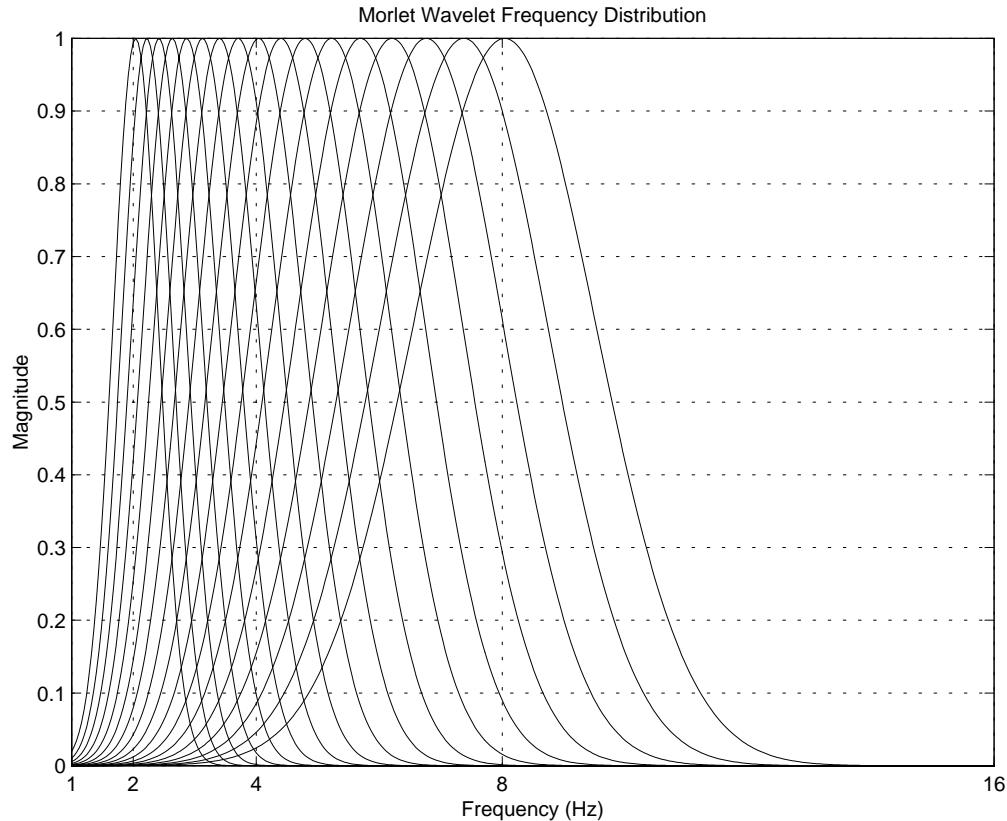
where:

M - the mean modulus of the Wavelet coefficients.

F - the center frequency at a given scale

PSD - Power Spectral Density

The following arbitrary plot is a frequency domain representation of the Morlet wavelet at 17 scales. The wavelets span 3 octaves. For perfect reconstruction, there are 8 voices or subdivisions between octaves totaling 17 scales. Again, it is evident that wavelets at higher frequencies have a wider passband resulting in extra energy when calculating PSD.



References

- (8.143), (11.146), (Table 9.6) Bendat, J. S. & Piersol, A. G. 1986 *Random data: analysis and measurement procedures*, 2nd edn. New York: Wiley.
- (2) Hesany, V., Moore, R. K., Gogineni, S. P., & Holtzman, J. C., "Slope-induced nonlinearities on imaging of ocean waves," *IEEE Journal of Oceanic Engineering*, vol. 16, no. 3, 1991